Code 582

Flight Software Branch

[MISSION NAME] MISSION

# [FSW SUBSYSTEM NAME] FLIGHT SOFTWARE REQUIREMENTS DOCUMENT

Flight Software Branch – Code 582

Template Version 0.8 – 06/19/06
582-2005-003
(Replace with Mission FSW Requirements Document Version)

Goddard Space Flight Center

Greenbelt, Maryland

National Aeronautics and
Space Administration

## DOCUMENT CONTROL

This document will be (is – after baseline) a [ MISSION NAME & ACRONYM – FSW SUBSYSTEM NAME & ACRONYM] Flight Software Team controlled document. Changes to this document require prior approval of the FSW Team Lead. Proposed changes shall be submitted to the [ MISSION ACRONYM – FSW SUBSYSTEM ACRONYM] FSW Configuration Control Board (CCB).

[FSW Team Lead Name]

[Title]

# FOREWORD

This document is a skeleton Requirements Document intended for use by Code 582 (Flight Software Branch) personnel as the basis for a mission-specific Flight Software Requirements Document.

The following style conventions are used throughout:

Text in this style (style name "BODY") is used for text that is equally applicable to all Requirements Documents and should be included without modification. All document section headings are in the same category (although their style names vary depending on outline level).

[Text in this style (style name "TAILORING ADVICE") is advice on how to tailor the text in any specific section.]

As the plan is developed, the generic [TAILORING ADVICE] text should be replaced with material that applies to the specific project.

## GENERAL TAILORING GUIDELINES

This section includes general tailoring guidelines applicable to the whole document. Specific recommendations are included in applicable sections.

All components of the table of contents should be addressed, but the level of detail is left up to the Team based on flight software complexity and customer needs/expectations. The length and level of detail of the document should be commensurate with the scope and complexity of the project. Section headings may be added where necessary, but existing headings should not be modified or deleted. If a particular section is not applicable to the specific document under production, that fact should be noted under the section heading, together with a brief explanation. Additional appendices may be added as necessary to provide supplementary information. Branch waiver is required for departure from this template.

The Flight Software (FSW) Requirements Document is the sole source for all of the requirements for the FSW Subsystem (i.e., the subject of this document). This is the document to be used by developers to decide what to build and by testers as the basis for developing their tests. Interface Requirements Documents (IRDs) will contain the interface requirements implied by the FSW Requirements Specification and will be referenced by this document. Other documents, such as Interface Control Documents (ICDs) or Users' Guides, may provide useful supporting information that describes the implementation details or operational contexts of the FSW Subsystem. However, they must not be the source of additional FSW requirements.

Update this document prior to launch to reflect the as-built requirements.

Use an automated requirements management tool in the development and maintenance of this FSW requirements specification. Many GSFC FSW projects have been using requirements management tools (e.g., Rational RequisitePro from IBM and DOORS from Telelogic, among others) to maintain a requirements database as well as the associated documentation (i.e., the subject of this template). The capabilities include not only requirements database management but also automatic document generation and requirements traceability support. A requirements document can be generated using the tool and the requirements database. Conversely, a requirements database can be built as the requirements document is being created. Certain contextual parts of the requirements document may have to be managed outside the tool and generated along with the document from database tool to generate the complete requirements document. .

Number requirements uniquely for traceability and identification purposes. No recommendation is made on a requirements numbering scheme. To avoid confusion, do not change or delete requirements numbers once they have been assigned; for deleted requirements change the requirement text to ‘Deleted’.

The following disclaimer appears on all pages: “Printed copies of this document are for REFERENCE PURPOSES ONLY! The only controlled copy of this document is located on-line at http://xxxxxxxx”. This disclaimer should be modified to contain the appropriate URL, but should not be removed.

Finally, in the target document, this entire section (“Foreword”) can be deleted or replaced with product-specific information.

## SIGNATURES

Submitted by:

___________________________________ ______________

Aaa A. Aaaaa/nnn — Date

[FSW SUBSYSTEM NAME] Subsystem Flight Software Lead

Approved by:

___________________________________ ______________

Elaine Shell/582 — Date

Flight Software Branch/Head

___________________________________ ______________

Aaa A. Aaaaa/nnn — Date

[Other Subsystem] Subsystem Flight Software Lead(s)

___________________________________ ______________

Aaa A. Aaaaa/nnn — Date

[ MISSION NAME] Mission Project Manager

___________________________________ ______________

Aaa A. Aaaaa/nnn — Date

[ MISSION NAME] Mission Systems Engineer

# UPDATE HISTORY

[This table shows the update history for the Requirements Document Template. For Requirements Documents developed using this template, the elements of this table should be replaced with mission-specific document update information.]

| Version | Date | Description | Affected Pages |
|---|---|---|---|
| 0.1 | 03/13/03 | Draft in progress – skeleton document created from Elaine's outline | All |
| 0.2 | 08/15/03 | Second version of draft – revised outline; content added | All |
| 0.3 | 09/11/03 | Formatting changes only – no content change | All |
| 0.4 | 09/18/03 | Third version of draft – content edited | All |
| 0.5 | 09/24/03 | Forth version of draft –content edited | All |
| 0.6 | 09/24/03 | Fifth version of draft – revised appendices; content edited | All |
| 0.7 | 05/08/06 | Revised by K. Narayanan. Combined and updated Functional and Performance Requirements. Removed Reliability Requirements. Minor changes to other sections.<br>Changes to close DCRs #101, 103. | All |
| 0.8 | 06/19/06 | Revised by K. Narayanan. Minor changes to sections 1.3 and 3.3 to close DCRs #161, 162. | 1, 6 |

## CONTENTS

# 1.0 INTRODUCTION

## 1.1 DOCUMENT PURPOSE

The [FLIGHT SOFTWARE SUBSYSTEM NAME AND ACRONYM] for the [MISSION NAME AND ACRONYM] mission will be developed by the Flight Software Branch (FSB) of the Information Systems Division (ISD). The purpose of this requirements specification is to define the requirements to be satisfied by the [FSW SUBSYSTEM ACRONYM] system.

## 1.2 DOCUMENT SCOPE

The scope of this document is limited to the specification of requirements for the [MISSION ACRONYM] [FSW SUBSYSTEM ACRONYM] software requirements. These include functional, performance, qualification, and design requirements. The requirements for communication interfaces between the [FSW SUBSYSTEM ACRONYM] subsystem and other software and hardware subsystems are documented in the appropriate Interface Requirements Documents (IRDs).

## 1.3 DOCUMENT ORGANIZATION

This document is organized into six additional sections and several appendices.

Section 2 gives the [FSW SUBSYSTEM ACRONYM] context.

Section 3 describes the [FSW SUBSYSTEM ACRONYM] external interfaces.

Section 4 documents the [FSW SUBSYSTEM ACRONYM] system design decisions and constraints.

Section 5 contains the [FSW SUBSYSTEM ACRONYM] functional and performance requirements.

Section 6 discusses the [FSW SUBSYSTEM ACRONYM] resource utilization and margins.

Section 7 contains the [FSW SUBSYSTEM ACRONYM] qualification strategy.

Appendix A contains a list of abbreviations and acronyms used in this document.

Appendix B contains requirements for failure detections and responses for the [FSW SUBSYSTEM ACRONYM].

Appendix C contains the Common Flight Software Executive requirements for the [FSW SUBSYSTEM ACRONYM].

## 1.4 RELEVANT DOCUMENTS

### 1.4.1 Parent Documents

Provide a list of Parent Documents (i.e., mission requirements, relevant subsystem requirements, IRDs, ICDs, hardware specifications, algorithm documents, etc.)

### 1.4.2 Reference Documents

Provide a list of Reference Documents (e.g., related requirements, Product Plan, Test Plan, CCSDS standards, technical documents, etc.)

# 2.0 SYSTEM CONTEXT

This section is provides a complete context for [FSW SUBSYSTEM ACRONYM] subsystem. The context provides the reference architecture, or framework, from which the FSW requirements are derived. The requirements are derived by developing the next level of detail in the reference architecture.

This section should briefly describe the system in sufficient detail to permit an understanding of the overall system in the context of its intended system interface and operations environments. It is intended to cause the author to think through the FSW to the level appropriate to establish placeholders for areas of requirements. The section should be only a few pages long but should give the reader a solid sense of where the FSW requirements will be coming from. Diagrams of the flight data system and mission FSW architecture may be useful.

## 2.1 MISSION OVERVIEW

Provide high-level descriptions of the mission including information categories such as: mission purpose, science instruments, orbit, Earth communications, and the ground system.

## 2.2 ON-ORBIT OPERATIONS ENVIRONMENT

Explain the role of the FSW described in this document during science operations, launch, deployment, ground communications, maintenance, and end-of-life. Describe, in conceptual terms, ground communications contacts, science plan execution, GN&C, CPU power state/role during launch, on-board calibrations, and on-orbit maintenance.

## 2.3 FLIGHT HARDWARE ENVIRONMENT

This section provides the context of the FSW in the flight hardware environment. Include a diagram of the flight data system and special purpose hardware that shows all flight hardware referenced in this document and where the FSW resides on the hardware.

### 2.3.1 Flight Data System

Identify the flight data system hardware environment of the FSW described in this document —what is the role of the FSW being described with respect to the flight hardware (e.g., memory types and how used by the FSW, data buses and whether this FSW has direct interface to the bus, etc.).

### 2.3.2 Special Purpose Hardware

Identify sensors, actuators, transponders, power elements, instruments, detectors, and any other special purpose hardware that the FSW described in this document must support.

## 2.4 FLIGHT SOFTWARE ENVIRONMENT

Identify other FSW subsystems that this FSW interfaces with, and what services each provide to the others. For example, if the subsystem of this document is ACS, it lives within C&DH FSW. If the subsystem is C&DH, it supports ACS as well as power, science instruments, etc.

## 2.5 GROUND SYSTEM OPERATIONS CONTEXT

This section should provide a view of the FSW from the ground system perspective. Identify the environments for ground system operations, FSW testing, and I&T. Include any ground communications and autonomy drivers to the extent that they influence the requirements defined within this document.

# 3.0 FSW EXTERNAL INTERFACES

This section provides high-level descriptions of the external interfaces for the [FSW SUBSYSTEM ACRONYM] subsystem. The intent is to identify the external interfaces that the [FSW SUBSYSTEM ACRONYM] subsystem software is dependent on or must support. These descriptions identify the set of communication activities between the [FSW SUBSYSTEM ACRONYM] subsystem and the external [MISSION ACRONYM] mission components identified in Section 2. Any [FSW SUBSYSTEM ACRONYM] subsystem software functional and performance requirements related to these interfaces are provided in Sections 5 or 6, respectively. The actual communication interface requirements for the [FSW SUBSYSTEM ACRONYM] subsystem external interfaces are specified in the following Interface Requirements Documents (IRDs): [provide IRD list].

Although there will be IRDs with interface requirements and ICDs going into elaborate detail about the external interfaces, a simple description of the types of content to be dealt with on each of the interfaces is useful for identifying the quantity of questions to be resolved for FSW functional requirements.

Provide a diagram of all external interfaces, and walk-through the interfaces one at a time, identifying both directions of data transfer separately. Address questions such as: where does time come from for this FSW, and what format is required? Where will commands be coming from, and are they all handled in real-time by individual tasks? Identify whether each external hardware device provides positive hardware status to every command. If all provide status, just say all. This is intended to be at the conceptual level only. This whole section should be 3-5 pages, maximum.

## 3.1 EXTERNAL HARDWARE INTERFACES

Describe flight hardware interfaces to the FSW subsystem. Include hardware elements such as processors, Attitude Control Electronics, Star Trackers, Solar Arrays, instrument controllers, etc. as appropriate for this particular FSW subsystem.

An example interface description for Attitude Control Subsystem (ACS) FSW and Star Tracker (ST) hardware is given below:

> "There are two digital star trackers. The ACS software will routinely use data from both STs for spacecraft attitude update and stellar acquisition. ST data transfers will occur via the ACS 1773 bus.
>
> Star Trackers-to ACS Software
>
> Both star trackers independently generate data at a 10 Hz rate. The data will be formatted into raw data packets and sent to the ACS Processor where the C&DH software will maintain a FIFO buffer for data from each ST. The ACS software will accept the ST data from the C&DH at a 1 Hz rate.
>
> ACS Software-to-Star Trackers
>
> The ACS software will command each ST with up to 5 reduced fields of view and star magnitude thresholds for routine star updates. In addition, the ACS software will command star tracker field of view maps in order to generate adequate star data for ground analysis of spacecraft pointing performance."

## 3.2 EXTERNAL SOFTWARE INTERFACES

Discuss issues and characteristics associated with other FSW subsystems. Describe interfaces such as the software bus, time distribution, operating system, commands, and telemetry, as appropriate.

An example interface description for ACS software and the ACS Command and Data Handling Software (C&DH) is given below:

> "The ACS C&DH software in the ACS processor is an extension of Spacecraft Processor C&DH functions. Stored command management, memory load and dump, table dump, processor fault checks, time code maintenance, and memory dwell telemetry generation are all provided to support the ACS software.
>
> <u>ACS C&DH Software-to-ACS Software</u>
>
> Software Bus: The C&DH software environment includes a "software bus" to which the ACS software must interface. All communications between the ACS software and its external interfaces are via the software bus. The ACS tasks will be scheduled for execution by the software bus and all internal exchange of data between tasks in the ACS processor is accomplished by the software bus. The ACS software has no direct access to the real-time operating system.
>
> Spacecraft Time: The C&DH software provides spacecraft time to the ACS software by insertion of Spacecraft Time (i.e., Mission Elapsed Time) into the Attitude Control Electronics (ACE) data packet prior to providing the ACE packet to the ACS software. The ACE packet time will represent the time at which the ACE data was sampled with respect to the 1 Hz clock signal.
>
> Universal Time Correlation Factor: The C&DH software will maintain a UTC correlation factor that, when applied to Spacecraft Time, will represent the accurate time by which ground planning and science data processing takes place. The ground will update the UTC correlation factor as appropriate to maintain its accuracy. The ACS software needs time in UTC format in order to interpret and process ephemeris data. The C&DH software will provide the UTC correlation factor to the ACS software.
>
> Requests for ACS Housekeeping: The C&DH software will request routine housekeeping telemetry from the ACS software according to a predefined repetitive scheme. Repeated lack of ACS software response to the status request will be considered indication that the ACS software task is no longer executing properly.
>
> ACS Input Commands: Commands from the ground, stored command memory, a C&DH software application or another ACS task will be provided to the ACS control task software via a command input queue. Each ACS software task will poll its command input queue once every ACS control cycle.
>
> Flight Hardware Data and Status: All data and status from the flight hardware is obtained by the C&DH bus scheduler software at a predefined deterministic rate over the ACS 1773 bus. All data is provided to the ACS software as CCSDS telemetry packets via dedicated input queue.
>
> <u>ACS Software-to- ACS C&DH Software</u>
>
> ACS Telemetry Output: All ACS telemetry (processed data, ACS software status and housekeeping, etc.) will be formatted as CCSDS packets (with a time stamp of the ACE packet received at the start of this ACS control cycle) and transferred to the Spacecraft Processor for immediate downlink and/or onboard data storage. The ACS software will generate all of its synchronous telemetry packets all of the time regardless of the active downlink bandwidth. The C&DH software will intercept ACS software telemetry for status monitoring purposes and will downlink/store data according to predefined filter specifications.

ACS Commands to Flight Hardware: All ACS commands to flight hardware on the ACS 1773 bus are issued as CCSDS telecommand packets to the C&DH software bus. The ACS C&DH bus scheduler outputs the commands onto the ACS bus at predefined time slots per ACS control cycle. The specification of ACS bus 1773 time slots, for telemetry from hardware and commands to hardware, is defined to support ACS processing performance requirements."

## 3.3 GROUND INTERFACES

Discuss interface issues and characteristics between ground operations and the FSW subsystem. Describe uplink command interfaces, at a high level (e.g., command routing in the context of the flight hardware and software elements). Provide a similar description for downlink telemetry. Identify any interface standards such as CCSDS for command and telemetry.

An example interface description for ACS software and ground operations is given below:

"Ground-to-ACS Software

Ground commands to the ACS Processor will have unique CCSDS application IDs. ACS commands put onto the 1773 Spacecraft Data Bus by the C&DH Uplink Card will be routed directly to the ACS Processor by the 1773 spacecraft bus controller. This includes ACS table and memory load commands.

ACS Software-to-Ground

ACS software does not interface directly to the ground for telemetry downlink."

# 4.0 DESIGN SPECIFICATIONS

The [FSW SUBSYSTEM ACRONYM] subsystem design specifications specify heritage software to be used and any other design restrictions imposed on the [FSW SUBSYSTEM ACRONYM] subsystem.

These restrictions could include heritage software; prescribed data system hardware, operating system, or software language; or even a prescribed software development methodology or standard (e.g., the use of the Unified Modeling Language – UML).

## 4.1 HERITAGE SOFTWARE DECISIONS

The FSB Common Flight Software Executive will be used as heritage C&DH software on the [MISSION ACRONYM] mission. The following Common Flight Software Executive functions will be used: [provide list].

The FSB Common Flight Software Executive requirements are documented in Appendix C and are a part of the requirements set for the [FSW SUBSYSTEM ACRONYM] subsystem software. Proposed changes to the FSB Core software must be submitted to the FSB Flight Software Reuse CCB. Approved changes will be provided to the [FSW SUBSYSTEM ACRONYM] subsystem software team by the CCB.

List any other heritage software mandated for use in the context of this FSW subsystem. If there are no others, say so.

Indicate how changes to additional heritage software will be documented and controlled.

## 4.2 DESIGN CONSTRAINTS

This section should address any restrictions or available features that impact the FSW design such as:

- Development Process & Tools (e.g, RUP, UML, MATLAB, etc.)
- Coding & Other Implementation Standard and Approaches (e.g., C++, static versus dynamic addressing, etc.)
- Communication Standards (e.g., CCSDS COP-1, CCSDS Packet Telemetry, CCSDS CFDP, IEEE Standards, ISO Standards, etc.)
- Data System Hardware (e.g., floating point hardware, memory protection hardware, etc.)

## 5.0 FUNCTIONAL AND PERFORMANCE REQUIREMENTS

This section gives the functional and subsystem-level performance requirements for the [FSW SUBSYSTEM ACRONYM] subsystem. Functional requirements may contain performance attributes (e.g., delays between intra-FSW subsystem components or loading requirements for individual subsystem functions) as part of their specification. Subsystem-level performance requirements specify mission performance and timing requirements attributed to the FSW subsystem.

Functional requirements specify what the system does in terms of: source, quantity, units of measure, timing and range of valid inputs; processing on inputs (i.e., validity checks, sequence of operations, response to abnormal situations); and destinations, quantities, units of measure, range of valid outputs, disposition of illegal inputs, and error messages. Functional requirements should include computational precision requirements of specific data elements that must meet certain criteria (e.g., hardware accuracies, dynamic model accuracies, computational accuracies, error accuracies, levels of precision, etc.) as appropriate. Include references to IRDs or ICDs as appropriate.

Subsystem-level performance requirements specify quantitative attributes of the FSW subsystem as a whole for static and dynamic conditions. Required levels of performance should be specified with respect to FSW subsystem loading allocated from system-level requirements, if applicable. Mission performance requirements for the FSW subsystem may include performance characteristics of attitude control, ephemeris propagation, pointing, orbit, gimbals, or ground contacts. Timing requirements specify timing characteristics such as response time and throughput.

Examples of subsystem-level performance requirements are:

> “(P) The ACS Sun Acquisition shall acquire a power positive orientation, to within 15 degrees of desired pointing within 10 minutes of ACS entering the Sun Acquisition mode.”
>
> “(P) The maximum slew rate allowed by the ACS Inertial Pointing shall be 5.0 deg/sec,per axis.”
>
> “(P) C&DH software shall provide UTCF to applications within 0.1 seconds.”
>
> “(P) ACS software shall output reaction wheel commands within 0.06 seconds.”
>
> “(P) The Common C&DH System shall take less than 10 seconds to start up. “

The [FSW SUBSYSTEM ACRONYM] subsystem functional and performance requirements were developed using the reference architecture and communication activities described in Sections 2 and 3 and then proceeding to the next level of detail. The [FSW SUBSYSTEM ACRONYM] subsystem was decomposed into its major functions, and the communication activities between those functions and the external interfaces were identified. This reference architecture provides the basis for the functional and performance requirements.

The FSW subsystem reference architecture and communication activities define a set of taxonomy that provides the scope, or level of detail, appropriate for the requirements. Requirements that use the taxonomy of the (previous) higher-level reference architecture will be too vague for the FSW requirements. Requirements that use a taxonomy of a yet-to-be-defined reference architecture will be too detailed.

The following paragraphs provide specific rules for writing functional requirements. Organizations for specific subsystem functional requirements are given at the end of this section.

Partition functional requirements according to the templates provided below for the specific FSW subsystem. A basic goal in establishing the top-level requirements categories is to cover all requirements in a consistent and complete hierarchy. The lower level decomposition provides the opportunity to refine the requirement categories to the appropriate level of detail. Lower levels of partitioning should facilitate readability via a logical flow of requirements.

Organize requirements with the expectation that modifications to each requirement area will be necessary. Structure, style and numbering conventions impact the ease of use and readability of changed requirements. Requirements should be structured and styled such that necessary changes can be made easily, completely, and consistently as requirements evolve.

Use a number along with a subsystem prefix to uniquely identify each requirement (e.g., CS1025, SB2134).

Use “(P)” before the requirement text to identify the performance requirements (i.e., to distinguish them from other requirement categories).

Identify child requirements with a hierarchical numbering scheme using the parent requirement number as a prefix (e.g., parent requirement number SSxxxx; children requirements SSxxxx.1,SSxxxx.2, etc.).

Do not re-number requirements if some are deleted. Leave the existing number and title, if applicable, and replace the text of the deleted requirement with the word “Deleted.”

Use language in writing requirements to make the requirements document easy for a non-FSW person (e.g., mission systems engineer) to understand. Use short declarative statements. Keep phrases functional instead of design. Requirements language should not be academic but should use concrete, FSW-specific terminology.

Provide action to each requirement. Use "The FSW shall …"; or "In the event of …, the FSW shall …". The word “shall” is reserved for the actual requirement. The word “will” is often used in the clarifying text.

Avoid writing compound requirements (i.e., multiple “shall” in the same requirement statement) when possible. For example, a requirement may specify a capability that gets reported in telemetry. The issue is whether to specify this as a compound requirement or as two separate requirements (i.e., one requirement for the required capability and another for the telemetry reporting). Create appropriate child requirements.

Keep the focal point on the FSW functionality – not the ground or an external interface's perspective. "The ground shall send …" does not belong in a FSW requirements document.

Write requirements to be unambiguous. Each requirement must have one and only one interpretation. The most common complaint of FSW requirements is they are too vague. Developers and testers must interpret a requirement to mean the exact same thing.

Ensure that each requirement represents a complete statement. Typically, a requirement should have a beginning, middle and end: a subject (typically the FSW), a “shall” verb, and a predicate (i.e., the object of the action).

Specify a requirement for every non-nominal situation that can be conceived. Include command errors, exceptions, I/O errors, mode transitions, etc. If something can go wrong, there must be a requirement that states how the FSW is to react to the problem.

Don't drive requirements down to a level more detailed than necessary. Volume of requirements or requirements that get into design do not indicate good requirements. Including information in notes or requirements statements such as low-level (i.e., lower than the reference architecture for these requirements) functional decomposition and data flow diagrams, or object class diagrams is not appropriate. Hardware design details (e.g., registers) are also not appropriate. Keep in mind the advice given above about a reference architecture in the general guidelines: Requirements that use a taxonomy of a yet-to-be-defined reference architecture will be too detailed.

Ensure that each requirement is testable/verifiable via execution of the flight code and resources available to the ground (e.g., telemetry, events, logs, etc.). If a test case cannot be defined for a requirement, there is usually something wrong with the wording of the requirement. In only rare instances, is code inspection an acceptable manner to verify FSW implementation of a requirement. Note that the highest-level requirements may be stated generically and may not be directly testable. However, their decomposition into lower level requirements must be testable.

Do not create redundant requirements. There is something wrong with the organization of requirements if there appears to be a need to state a requirement twice or differently in another section of the document. However, it may be desirable in some cases to repeat a requirement category for different conditions. For example, there are a number of requirements that are present in several control modes. In this case, mode entrance and exit conditions could appear twice, as an entrance condition for one mode and an exit condition for the other mode. It may be desirable to state requirements separately in each control mode in which they apply.

Use diagrams and tables when appropriate (e.g., mode transitions, timing relationships of tasks, control logic, etc.). Ensure that each capability of the FSW is summarized in requirements terminology referencing the diagram or table.

Provide supplementary text and notes preceding the statement of the requirement to clarify the characteristics, scope, intent, rationale, operations implications, etc. of the requirements. This supplementary text must be clearly separated from numbered requirements. Notes are strongly encouraged when they clarify requirements' decisions. Notes are meant to be informational only, not part of the formal requirement. Note that this information can be captured as requirements attributes in requirements management tools.

Use a requirements management tool to maintain functional requirements and their associated attributes, such as rationale. Both the requirements and their attributes can be extracted from the tool for inclusion in this document using the typical automated document generation capabilities in these tools.

Provide definitions of all terms and units of measures where appropriate. Include a glossary of terms if necessary.

Use the same terminology throughout the document. Do not permit terminology or word usage to change from one part of a system's requirements to another. There must be no conflict between requirements (e.g., different characteristics specified for the same thing or logical or temporal conflict).

Use terminology according to the definitions in FSW Branch Glossary. Project-unique terminologies, which are not under the control of the FSW team, should be provided to the FSW CCB CMO for consideration. Project-unique definitions must be clearly stated as Project-unique in the requirements document.

Ensure the traceability of all requirements. Origins of requirements must be clear. Requirements must have backward and forward traceability. Use the requirements management tool to maintain the traceability. See Section 7 for traceability requirements.

Use the following organization for ACS FSW functional requirements:

5.0 ACS FSW Functional Requirements

5.1 Initialization and Separation Transitions

5.2 Sensor Data Processing

5.3 Orbit Model Maintenance

5.4 Spacecraft Attitude Determination and Gyro Drift Bias Determination

5.5 Spacecraft Attitude Control

5.6 Actuator Command Generation

5.7 Antenna Command Generation

5.8 ACS Redundant Component Management

5.9 Failure Detection and Handling

5.10 Telemetry Generation

5.11 Command Processing

Use the following organization for mission-specific C&DH FSW functional requirements:

5.0 Mission-specific C&DH Functional Requirements

The FSB Common Flight Software Executive provides common C&DH services. These requirements are included in Appendix C and are considered to be part of the [FSW SUBSYSTEM ACRONYM] subsystem C&DH functionality requirements. The core set of services from the Common Flight Software Executive (i.e., software bus, executive and task services, operating system abstraction layer, table management, time management, exception handler, and event manager) requires only mission-specific configuration parameter values to be specified.

5.1 Software Bus Configuration Parameters

5.2 Executive and Task Services Configuration Parameters

5.3 Operating System Abstraction Configuration Parameters

5.4 Table Management Configuration Parameters

5.5 Time Management Configuration Parameters

5.6 Exception Handler Configuration Parameters

5.7 Event Handler Configuration Parameters

5.8 Telemetry Management

5.9 Command Management

5.10 Telemetry Monitoring, Checking, and Response

5.11 Recorder Management

5.12 Stored Commanding

5.13 Data System Health and Safety

5.14 File Management

5.15 Memory Management

5.16 External Bus Driver

5.17 Local Bus Driver

5.18 Bootstrap Loader

5.19 Memory Read/Write

5.20 1553 Driver

5.21 Time Driver

Use the following organization for Power Subsystem FSW functional requirements:

5.0 Power Subsystem FSW Functional Requirements

5.1 Initialization

5.2 Commanding

5.3 Control Requirements

5.4 Failure Detection and Correction

# 6.0 RESOURCE UTILIZATION

This section specifies the subsystem-level resource utilization and margins for the [FSW SUBSYSTEM ACRONYM] subsystem.

Specify utilization, margin, and rate requirements for CPU, memory (RAM, PROM, and EEPROM), and bus hardware (e.g., PCI, 1553, Spacewire) used by the FSW subsystem. Utilization metrics may be specified as a percentage of bandwidth (i.e., execution rate); as a percentage of time; or as a percentage of space in the case of memory. Alternatively, the metrics may be specified in the actual units (e.g., bytes, execution rate, or transfer rate) of the resources. Margins may be appropriate for the utilization levels for contingency purposes. The resource metrics shall satisfy the latest version of the GSFC's "Gold Rules" in: http://pows002.gsfc.nasa.gov/aetdmetrics/designRules/DesignRulesFinal.pdf. See Table below for an example.

| **Mission Phase** | **SRR** | **PDR** | **CDR** | **Ship/Flight** |
|---|---|---|---|---|
| **Method** | **Estimate** | **Analysis** | **Analysis/Measured** | **Measured** |
| Average CPU | 50% | 50% | 40% | 30% |
| CPU Deadlines | 50% | 50% | 40% | 30% |
| PROM | 50% | 30% | 20% | 0% |
| EEPROM | 50% | 50% | 40% | 30% |
| RAM | 50% | 50% | 40% | 30% |
| PCI Bus | 75% | 70% | 60% | 50% |
| 1553 Bus | 30% | 25% | 20% | 10% |
| Spacewire (1355) | TBD | TBD | TBD | TBD |
| UART/Serial I/F | 50% | 50% | 40% | 30% |

Table: C&DH Flight Software Margins

# 7.0 QUALIFICATION STRATEGY

This section presents the strategy for the qualification of the [FSW SUBSYSTEM ACRONYM] subsystem software for use in development and test laboratories and ultimately in flight aboard the spacecraft. Test strategy and test environment required for validation are detailed along with the requirements traceability approach.

## 7.1 TEST STRATEGY

The [FSW SUBSYSTEM ACRONYM] subsystem software qualification requires a thorough test strategy designed to detect errors and deficiencies as early as possible while ensuring proper inter-operation of components integrated into subsystems and systems. To accomplish this, multiple levels of testing and test types are defined to insure a robust test process.

Test levels will include unit, build integration, build verification, and FSW system validation/acceptance.

Test types will include functional, performance, interface, load/stress, long duration, and regression .

## 7.2 TEST ENVIRONMENT

Specify any specifications for the FSW test environment. The FSW development team is responsible for unit testing. Unit tests may take place on the same desktop platform used for development. For example, if the development methodology involves using UML models, unit testing can be conducted on the same platform used for development by using the UML simulation capability. Any requirements such as these should be stated here.

Specify any unique test tools, equipment, and facilities that must be developed or available during FSW testing. Test levels beyond the Unit test level require increased levels of fidelity in the representation of the flight elements external to the FSW as well as the target execution environment for the FSW itself. Specify the test tools, equipment, and facilities requirements for each test level, as appropriate. The test environment could include the following:

- Flight-equivalent FSW hardware platform environment, including processors, buses, and operating system
- Flight component simulators
- Actual flight hardware and software components
- Ground system or proxy for command and telemetry processing
- Monitoring and test tools

### 7.3 REQUIREMENTS TRACEABILITY

A requirements traceability matrix will be maintained to trace the [FSW SUBSYSTEM ACRONYM] subsystem requirements.

Provide a mapping of the system requirements from the applicable high-level system specifications to the engineering requirements in this document. This traceability matrix should verify that all relevant requirements in the parent documents have been incorporated into the requirements in this document. This matrix will also include traceability of the requirements to the design specifications and the test plans.

Note that various requirements management tools provide capabilities to trace requirements and document the traces (e.g., traceability matrix). The matrix could be generated and included in this document automatically from the tool, including any changes due to requirements modification as the requirements evolve.

Use the automated requirements management tool to support traceability in the following manner:

- Enter the system-level requirements into an automated database in the tool if such a database does not already exist, or import an existing database into the tool if possible.

- As FSW subsystem requirements are recorded in the tool database, map these FSW requirements to the system-level requirements.

- Use the FSW requirements database as the basis for developing, documenting, and mapping the next level of FSW requirements, design specifications, and test plans.

- Produce requirements traceability matrix and other documentation for inclusion in this requirements specification directly from the tool.

- Ensure that the different levels of requirements are maintained and controlled.

# APPENDIX A ACRONYMS AND ABBREVIATIONS

This appendix contains the list of abbreviations and acronyms for the [MISSION ACRONYM] mission and the [FSW SUBSYSTEM ACRONYM] subsystem used in this document. These abbreviations and acronyms adhere to FSB standard usage.

# APPENDIX B FAILURE DETECTIONS AND RESPONSES

This appendix identifies requirements for all failure detections and responses for the [FSW SUBSYSTEM ACRONYM] subsystem.

These are flight requirements for actions associated with event detections in the software. These requirements will be developed over time. Testing of these requirements will often require reconfiguration of the FSW to accommodate an action. This could be a separate document and must be a living document all through development and pre-launch.

# APPENDIX C COMMON FLIGHT SOFTWARE EXECUTIVE REQUIREMENTS

This appendix contains the FSB Common Flight Software Executive functional requirements. These requirements are maintained and controlled by the FSB. The controlled version of these requirements is located at [give URL]. This appendix will be updated after each approved version is published.

Insert the controlled versions of the FSB Flight Executive requirements in this appendix.

Insure that these requirements are current. Update this appendix when the requirements are updated.